/ CONFIG

#pragma config FOSC = HS // Oscillator Selection bits (HS oscillator)

#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT disabled)

#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)

#pragma config BOREN = ON // Brown-out Reset Enable bit (BOR enabled)

#pragma config LVP = ON // Low-Voltage (Single-Supply) In-Circuit Serial Programming Enable bit (RB3/PGM pin has PGM function; low-voltage programming enabled)

#pragma config CPD = OFF // Data EEPROM Memory Code Protection bit (Data EEPROM code protection off)

#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Write protection off; all program memory may be written to by EECON control)

#pragma config CP = ON // Flash Program Memory Code Protection bit (All program memory code-protected)

// #pragma config statements should precede project file includes.

// Use project enums instead of #define for ON and OFF.

#include <xc.h>

#define \_XTAL\_FREQ 20000000 // Set your crystal frequency

#define PCF8574\_ADDRESS 0x40 // Write Address for PCF8574

// I2C Functions

void I2C\_Init() {

TRISC3 = 1; // Set SCL (RC3) as input

TRISC4 = 1; // Set SDA (RC4) as input

SSPCON = 0x28; // Enable I2C Master mode

SSPCON2 = 0x00;

SSPADD = ((\_XTAL\_FREQ / 4) / 100000) - 1; // 100kHz I2C speed

SSPSTAT = 0x00;

}

void I2C\_Start() {

SEN = 1; // Start condition

while (SEN);

}

void I2C\_Stop() {

PEN = 1; // Stop condition

while (PEN);

}

void I2C\_Write(unsigned char data) {

SSPBUF = data;

while (!SSPIF); // Wait for completion

SSPIF = 0;

}

unsigned char I2C\_Read(unsigned char ack) {

RCEN = 1;

while (!BF);

unsigned char data = SSPBUF;

ACKDT = ack;

ACKEN = 1;

return data;

}

// PCF8574 Write Function

void PCF8574\_Write(unsigned char data) {

I2C\_Start();

I2C\_Write(PCF8574\_ADDRESS);

I2C\_Write(data);

I2C\_Stop();

}

// PCF8574 Read Function

unsigned char PCF8574\_Read() {

unsigned char data;

I2C\_Start();

I2C\_Write(PCF8574\_ADDRESS | 1);

data = I2C\_Read(0);

I2C\_Stop();

return data;

}

void main() {

TRISB0 = 1; // Push button input (Active LOW)

TRISB1 = 0; // LED output

TRISB2 = 1; // Switch input

RB1 = 0; // Turn off LED initially

I2C\_Init();

while (1) {

unsigned char door\_status = (RB0 == 0) ? 1 : 0; // Read button (LOW = Door Open)

unsigned char switch\_state = RB2; // Read switch

// LED Logic

if (door\_status || switch\_state) {

RB1 = 1; // Turn ON LED

PCF8574\_Write(0x02); // P1 = 1 (LED ON)

} else {

RB1 = 0; // Turn OFF LED

PCF8574\_Write(0x00); // P1 = 0 (LED OFF)

}

\_\_delay\_ms(200);

}

}